its finite-precision numeric format. Since the constant does not change, one can design and use a constant multiplier which has much lower cost than a corresponding general multiplier. The price for the reduced cost is that the constant multiplier is not as flexible as a general multiplier. However, if one of the numbers being multiplied is known in advance, a dedicated constant multiplier can lead to low-cost transform implementations.

Constant multipliers and techniques for designing constant multipliers appear in US Patent 6,223,197 (issued to K. Kosugi on April 24, 2001), in US Patent 5,903,470 (issued to A. Miyoshi and T. Nishiyama on May 11, 1999), in US Patent 5,841,684 (issued to K. Dockser on November 24, 1998), in US Patent 5,815,422 (issued to K. Dockser on September 29, 1998), in US Patent 5,600,569 (issued to T. Nishiyama and S. Tsubata on February 4, 1997) and in US Patent 5,159,567 (issued to J. Gobert on October 27, 1992).

Another technique aimed at reducing the cost of a multiplication operation in signal processing transforms appears in the patent application NON-CONSTANT REDUCED-COMPLEXITY MULTIPLICATION IN SIGNAL PROCESSING TRANSFORMS by the inventor of the present invention. A non-constant, non-general multiplier is proposed. At least one of the numbers to be multiplied is not a constant and is also not allowed to take on any value permitted by its finite-precision numeric format. A non-constant, non-general multiplier can exploit restrictions on one or both numbers being multiplied to achieve greater flexibility than a constant multiplier at lower cost than a general multiplier.

It has long been recognized that there are a number of low-cost multiplication operations. For instance, multiplication by 1 has zero cost. In signed binary representations, multiplication by -1 can be as simple as flipping a sign bit. In binary twos complement representations, multiplication by -1 can be accomplished by flipping bits in a number representation and adding a constant number value.

Another low-cost multiplication operation in binary representations with power-of-two element values is shifting. Shifting a number represented in such a format implements multiplication by a power of two, such as multiplication by 2, by 4, or by 8, or multiplication by 1/2, by 1/4, or by 1/8.

Constant multipliers, non-constant, non-general multipliers, and low-cost multiplication techniques have been proposed for used in reducing the implementation cost of signal processing transforms such as the discrete Fourier transform, the discrete cosine transform, and the discrete sine transform, where transform weights are known constants. These techniques do not reduce the cost as measured by the number of multiplication operations. The do reduce the cost of individual multiplication operations. The cost reductions depend on exploiting special properties of the number values used in the transform, special properties of the finite-precision numeric formats in which those number values are represented, or special properties of the representations of the number values.

Fast Fourier transform techniques for computing discrete Fourier transforms have managed to reduce the complexity of these transforms from on the order of N^2 multiplication operations to on the order of N log N operations, where N is the transform size and the logarithm is of base 2. The order N^2 complexity is obtained from direct computation of a common closed-form expression of discrete Fourier transforms, and represents N inputs each multiplied by N weights, with the resulting products added to produce N outputs.

The order N log N complexity depends on special properties of discrete Fourier transform weights and lowest complexity is obtained for particular N values that are highly composite. A highly composite N value is one that can be factored into the product of very small integer factors. In most fast Fourier transform techniques, a large-sized transform is decomposed into a set of smaller-

sized transforms. It may be possible to recursively decompose the smaller-sized transforms to the level of very small integer factors.

At the lowest level of recursive decomposition, the transforms may be able to exploit some of the low-cost multiplication operations discussed above, as well as low-cost complex operations such as multiplication by purely real or purely imaginary numbers.

The weights in discrete Fourier transforms are complex numbers uniformly spaced on a unit circle in the complex plane. One of the weights is on the positive real axis. In a Cartesian complex coordinate system, discrete Fourier transform weights allow repeated application of the distributive property of multiplication. Fast Fourier transform techniques take advantage of this to compute discrete Fourier transforms using a sequence of computation stages. Each stage produces outputs that are weighted inputs. Intermediate stages compute intermediate outputs which contain non-zero net weighting of more than one transform input. Each intermediate output is passed to more than one input of the next stage.

Part of the multiplication complexity reduction of fast Fourier transform techniques and fast techniques for other transforms such as discrete cosine transform and discrete sine transforms is a result of shared computation of the products which appear in the closed-form direct expressions of the transforms. This complexity reduction depends on properties of the number values and on computation in Cartesian coordinates. This complexity reduction does not depend on special properties of the representations of the number values and also does not depend on special properties of particular finite-precision numeric formats.

Another example of shared multiplication complexity is embodied by an extension of the complex multiplier of US Patent 4,354,249 issued to T.M. King and S.M. Daniel on October 12, 1982. The complex multiplier is capable of